

Микропроцессорная архитектура

процессорное ядро МК
и его архитектура

Параметры МП, проц. ядра МК

- Разрядность процессора (максимальная разрядность данных с которыми могут выполняться операции)
- Архитектура ядра
- Набор команд (определяет принадлежность к семейству)
- Тактовая частота :
 - ✓ ядра,
 - ✓ обмена с ОЗУ, обмена с шиной [МГц, ГГц]
- Ширина шины данных и команд (может быть больше разрядности процессора, например у CM3 проц 32 бит а шина ОЗУ 128 бит),
- Интерфейс ОЗУ (DDR3,DDR4...), скорость чтения ОЗУ (Мб/сек)
- Производительность:
 - ✓ IPS, MIPS, GIPS... –число целочисленных инструкций в секунду,
 - ✓ FLOPS, - инструкций с плав запятой в секунду MFLOPS, GFLOPS, Tflops, Pflops
- Показатель производительности MFLOPS/МГц MIPS/MHz 0,01..... 1000
- Энергоэффективность Гфлопс/Вт
- Потребление= тепловая мощность рассеивания (КПД МП и МК всегда=0)
- Тип корпуса (определяет размеры, массу, тип выводов, технологию монтажа)

Производительность суперкомпьютеров

Производительность суперкомпьютеров

Название	год	FLOPS
флопс	1941	10^0
<u>килофлопс</u>	1949	10^3
<u>мегафлопс</u>	1964	10^6
<u>гигафлопс</u>	1987	10^9
<u>терафлопс</u>	1997	10^{12}
<u>петафлопс</u>	2008	10^{15}
<u>эксафлопс</u>	—	10^{18}
<u>зеттафлопс</u>	—	10^{21}
<u>йоттафлопс</u>	—	10^{24}
<u>ксерафлопс</u>	—	10^{27}

Компьютер **ЭНИАК**, построенный в **1946 году**, при массе 27 т и энергопотреблении 150 кВт, обеспечивал производительность в 300 флопс

- **IBM 709 (1957)** — 5 кфлопс
- **БЭСМ-6 (1968)** — 1 Мфлопс (операций деления)
- **Cray-1 (1974)** — 160 Мфлопс
- **БЭСМ-6 на базе Эльбрус-1К2 (1980-х)** — 6 Мфлопс (операций деления)
- **Эльбрус-2 (1984)** — 125 Мфлопс
- **Cray Y-MP (1988)** — 2,3 Гфлопс
- **Электроника СС БИС (1991)** — 500 Мфлопс
- **ASCI Red (1993)** — 1 Тфлопс
- **Blue Gene/L (2006)** — 478,2 Тфлопс
- **Jaguar (суперкомпьютер) (2008)** — 1,059 Пфлопс
- **IBM Roadrunner (2008)** — 1,105 Пфлопс [7]
- **IBM Sequoia (2012)** — 20 Пфлопс [8]

Российские супер ЭВМ

Вычислительные комплексы России	Эльбрус 3М	Эльбрус 4	Эльбрус 5	Эльбрус 6	Эльбрус 7
Год выпуска	2007	2011	2013	2016	2019
Производительность процессора (Гфлп)	4,8	19,2	64	256	768
Количество процессоров на сервере	2	16	64	64	64
Производительность сервера (Гфлп)	9,6	300	4000	16000	49000
Количество серверов в комплексе 64					
Производительность комплекса (Тфлп)	0,6	19	256	1000	3000

Персональные компьютеры на базе **МП**

- IBM PC/XT (1983) — 6,9 кфлопс
- ПК на основе процессора Intel 80386 (1985) с тактовой частотой 40 МГц — 0,6 Мфлопс
- Intel Pentium 75 МГц (1993) — 7,5 Мфлопс
- Intel Pentium II 300 МГц (1997) — 50 Мфлопс
- Intel Pentium III 1 ГГц (1999) — 320 Мфлопс
- AMD Athlon 64 2,211 ГГц (2003) — 840 Мфлопс[9]
- Intel Core 2 Duo 2,4 ГГц (2006) — 1,3 Гфлопс

Российские МП

Название МП	Эльбрус	Эльбрус 2С	Эльбрус 4С	Эльбрус 8С	Эльбрус 16С
Год выпуска	2007	2010	2012	2015	2018
Техн. норма (нм)	130	90	65	45	32
Частота (Мгц)	300	600	1000	2000	3000
Производительность МП (Гфлп)	4,8	19,2	64	256	768
Мощность (Вт)	6	16	25		

Эльбрус vs INTEL

Процессор, № (Имя)	Число ядер	GFlops, с ДТ	Частота, GHz	L3 cache, MB	Техпроцесс, нм	Тип ОЗУ	ОЗУ MAX	Слотов ОЗУ
Core i7 975	4	50	3.3	8	45	DDR3-1066	24	3
Эльбрус-4С (E2S)	4	50	0.8	0	65	DDR3-1600	48	3
2x Xeon x5677	4	104	3.5	12	32	DDR3-1333	288	9
i5-2500K (Sandy Bridge)	4	118	3.3	6	32	DDR3-1333	32	2
Эльбрус-8С (P1)	8	125	1.3	16	28	DDR3-1600	64	8
i7-4960X (Ivy Bridge)	6	244	3.6	16	22	DDR3-1866	64	4
i7-6950X (Broadwell E)	10	247	3.0	26	14	DDR4-3000	128	4
Эльбрус-8С2 (P9)	8	288	1.6	16	28	DDR4-2600	2048	8
i7-5960X (Haswell)	8	350	3.5	20	22	DDR4-2400	128	4
AMD A10-7850K	12	427	3.7	0	28	DDR3-2133	64	4
Эльбрус-16С	16	750	2.0	32	16	DDR4-2600	2048	8
Xeon E7-8890 v4	24	844	2.2	60	14	DDR4-1600	3078	12
Xeon E7-8894 v4	24	921	2.4	60	14	DDR4-1600	3078	12
Эльбрус-32С	32	2000-4000	2.0	≥32	14	DDR4-3200	2048	8
Xeon E5-2699 V5	32	2500	2.1	45	14	DDR4-3200	3078	12
Xeon Phi™ Processor 7210	72	3000	1.5	36	14	DDR4-2400	3078	12

Повышение производительности

1. Повышение тактовой частоты возможно за счёт: $E=C*U^2$; $Q=I\Delta t=I/f$; $f=I/Q=I/(C*U)$
 - уменьшения нормы проектирования 90,60,35,25, 16нм. (Повышается скорость перезаряда паразитной емкости). Сейчас разработка 10 нм технологии. В России освоение 16нм.
 - Снижения напряжения питания а для сохранения помехоустойчивости применение LVDS сигналов. Сейчас Уядра=1.8.... 1,2 В
 - Увеличения токов, но это повышает нагрев
2. Применение системных архитектурных решений

Архитектура фон Неймана

- Одна память для команд и данных (одна шина памяти)
- Современные Intel, AMD x86, x64 имеют архитектуру фон Неймана , хотя последние модели МП имеют встроенный контроллер ОЗУ и несколько шин к ОЗУ (для разных банков) и шину для периферии с целью ускорения при параллельной работе с разными банками памяти, но в них нет функционального разделения памяти на память данных и программ(кода инструкций)

Принципы фон Неймана

1. Принцип двоичного кодирования- вся информация кодируется в двоичном виде. (до Неймана ЭВМ работали в десятичных кодах)
2. Принцип однородности памяти. Каждая ячейка может быть как данным так и командой.
3. Принцип адресности. Ячейки памяти имеют адрес и значение. В произвольный момент доступна любая ячейка
4. Принцип программного управления (возможность условных и безусловных переходов)

Все вычисления, предусмотренные алгоритмом решения задачи, должны быть представлены в виде программы, состоящей из последовательности управляющих слов — команд. Каждая команда предписывает некоторую операцию из набора операций, реализуемых вычислительной машиной. Команды программы хранятся в последовательных ячейках памяти вычислительной машины и выполняются в естественной последовательности, то есть в порядке их положения в программе. При необходимости, с помощью специальных команд, эта последовательность может быть изменена. Решение об изменении порядка выполнения команд программы принимается либо на основании анализа результатов предшествующих вычислений, либо безусловно

Узкое место- доступ к памяти

Возможна самомодификация программ.

Первая ЭВМ с архитектурой фон Неймана- EDVAC, анонсирована в 1945г одновременно с **принципами** её построения

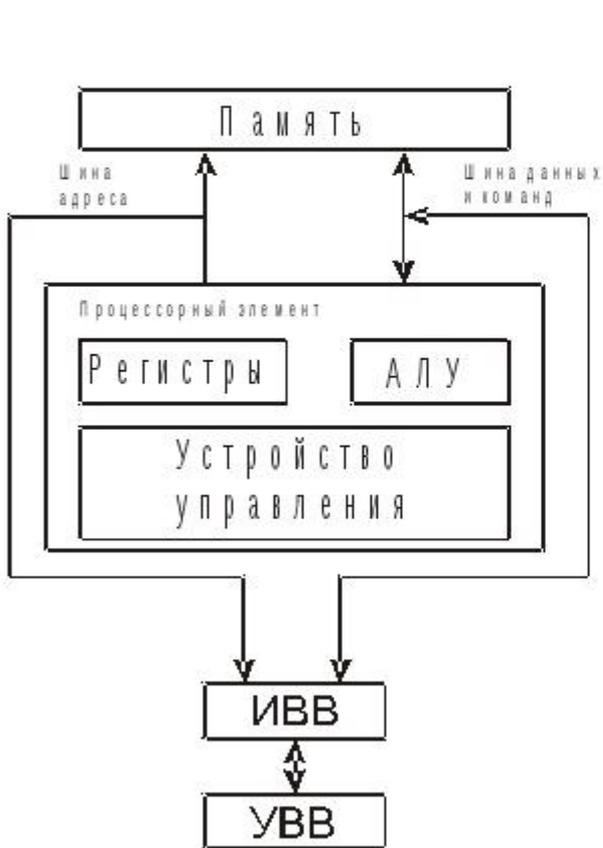
Гарвардская архитектура

- Разделение ОЗУ (ПЗУ) программ и ОЗУ данных. Одновременно можно читать/писать данные и читать следующую команду
- Современные варианты таких процессоров могут иногда содержать встроенные контроллеры сразу нескольких разнотипных шин для работы с различными типами памяти — например, ОЗУ, Flash (ПЗУ), регистры функциональных и периферийных модулей (ФМ).
- МК имеют обычно ядро с гарвардской архитектурой и несколько шин, Это легко реализовать, т.к. все шины, ПЗУ, ОЗУ, ФМ внутри кристалла и нет необходимости выводить шины на выводы МК

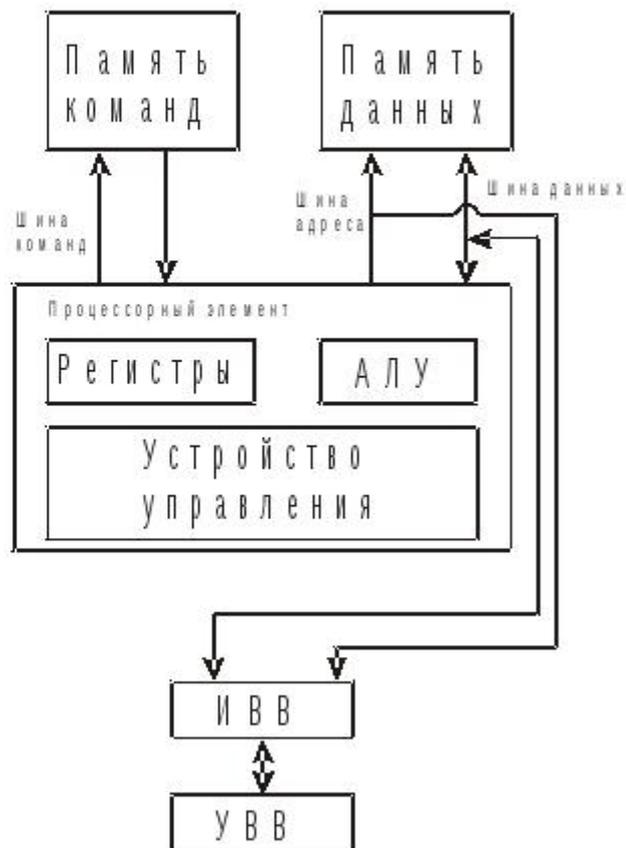


Архитектура микропроцессора

ОСНОВНЫЕ ТИПЫ



фоннеймановская архитектура



гарвардская архитектура

Архитектура по набору инструкций

- CISC,
- RISC,
- MISC

CISC архитектура

- *CISC — Complex Instruction Set Computer.*
- **Процессор со сложным набором инструкций**
- CISC отличается малым количеством регистров общего назначения-в Intel –только один регистр-А (акумулятор) с которым можно выполнять арифметические операции, большим количеством машинных команд (откуда следует и название). Это приводит к усложнению декодирования инструкций, что в свою очередь приводит к расходованию аппаратных ресурсов.
- Слабость CISC архитектуры заключается в том, что 80 процентов вычислений процессора приходилось на 20 процентов команд

RISC

- **Процессор с упрощённым набором инструкций**, англ. *RISC — Reduced Instruction Set Computer*. В этой архитектуре значительно более простое устройство управления. Большинство инструкций RISC-процессора содержат одинаковое малое число операций (1, иногда 2-3), а сами командные слова в подавляющем числе случаев имеют одинаковую ширину (PowerPC, ARM), хотя бывают исключения (Coldfire). У суперскалярных процессоров — простейшая группировка инструкций без изменения порядка исполнения.

Дейв Паттерсон и Карло Секуин сформулировали 4 основных принципа RISC:

- 1. Любая операция должна выполняться за один такт, вне зависимости от ее типа.
- 2. Система команд должна содержать минимальное количество наиболее часто используемых простейших инструкций одинаковой длины.
- 3. Операции обработки данных реализуются только в формате “регистр–регистр“ (операнды выбираются из оперативных регистров процессора, и результат операции записывается также в регистр; а обмен между оперативными регистрами и памятью выполняется только с помощью команд чтения/записи).
- 4. Состав системы команд должен быть “ удобен “ для компиляции операторов языков высокого уровня.

MISC

- **Процессор с минимальным набором инструкций**, англ. *MISC — Minimal Instruction Set Computer*. Эта архитектура определяется прежде всего сверхмалым количеством инструкций (несколько десятков), и почти все они нуль-операндные. Такой подход даёт возможность очень плотно упаковать код, выделив под одну инструкцию от 5 до 8 бит. Промежуточные данные в таком процессоре обычно хранятся на внутреннем стеке, и операции производятся над значениям на вершине стека. Эта архитектура тесно связана с идеологией программирования на языке Forth и обычно используется для исполнения программ, написанных на этом языке.

Современные CISC процессоры с SIMD и MIMD расширениями:

- CISC снаружи (для программиста существует возможность использовать много разнообразных команд процессора)
- и RISC внутри (все CISC команды выполняются внутри RISC набором **микрокоманд**)

EPIC архитектура

- **Процессор с явным параллелизмом**, англ. *EPIC* — *Explicitly Parallel Instruction Computer* (-ing, термин[®] Intel, HP). Отличается от прочих прежде всего тем, что последовательность и параллельность исполнения операций и их распределение по функциональным устройствам явно определены программой. Такие процессоры могут обладать большим количеством функциональных устройств без особого усложнения устройства управления и потерь эффективности. Обычно такие процессоры используют широкое командное слово, состоящее из нескольких слогов, определяющих поведение каждого функционального устройства в течение такта.

Параллельные архитектуры *SIMD/MIMD*

- **SIMD** (англ. Single Instruction, Multiple Data) — принцип компьютерных вычислений, позволяющий обеспечить параллелизм на уровне данных. Одной командой можно обработать много данных SIMD (например массив видео или аудио потока)
- **SIMD** и **MIMD** расширения увеличивают CISC набор инструкций процессора

	Одиночный поток команд (Single Instruction)	Множество потоков команд (Multiple Instruction)
Одиночный поток данных (Single Data)	<u>SISD</u>	<u>MISD</u>
Множество потоков данных (Multiple Data)	SIMD	<u>MIMD</u>

SIMD-расширения, используемые в процессорах архитектуры x86

- **MMX** — Multimedia Extensions. Коммерческое название дополнительного набора инструкций, выполняющих характерные для процессов кодирования/декодирования потоковых аудио/видео данных действия за одну машинную инструкцию. Впервые появился в процессорах Pentium MMX. Затем **MMX Extended**
- **3DNow! 3DNow! Extended-** от AMD
- **SSE, SSE2, SSE3,**
- **SSSE3** (в процессорах Intel Core 2 Duo.)
- **SSE4** Анонсирован 27 сентября 2006 года
- **AVX** SIMD-расширения Intel, которая представлена в 2010

Основные регистры процессорного ядра:

- PC-Счётчик (указатель) инструкций- содержит адрес следующей инструкции, при чтении команды автоматически увеличивается
- SP-Указатель стека (содержит адрес вершины стека в ОЗУ)
- Регистр состояния- содержит биты состояния ядра после последней выполненной команды (для ветвления, уровня прерываний и т.п.),
- Регистры общего назначения (РОН)- Универсальные (рабочие) регистры
- Аккумулятор (не во всех архитектурах)- спец регистр для операций.
- Регистры указателей (не во всех архитектурах)- спец регистры, в них может быть записан только адрес операнда для операций с памятью чтение/запись и инструкций с косвенной адресацией

В современных МП и МК в ядре может быть **несколько наборов регистров (регистровых банков)**: для системы, для приложения, подпрограмм обработки прерываний, для упреждающего выполнения с ветвлением. Это позволяет не сохранять регистры в стеке при переходе в прерывания, от приложения к системе и обратно, значительно экономя время.

Архитектура по регистрам в ядре

- Регистровая архитектура
- Аккумуляторная архитектура
- Стековая архитектура

Регистровая архитектура

- характеризуется свободным доступом к регистрам для выборки всех аргументов и записи результата. Элементарны арифметико-логические операции в таких процессорах кодируются в двух-, или трёхоперандные инструкции (регистр+регистр→регистр, иногда регистр результата совпадает с источником одного из аргументов).
- Все регистры кроме PC, SP и состояния равноправны и могут содержать операнд, адрес операнда
- Чем больше регистров, тем быстрее процессор и короче код. В регистрах (а не в ОЗУ) можно размещать локальные переменные, например счётчики циклов (счётчики i , j и т.п)
- Характерна для RISC процессоров.

Аккумуляторная архитектура

- **Аккумуляторная архитектура** — из регистров выделяется один - аккумулятор (или несколько регистров-аккумуляторов). Регистр-аккумулятор является источником одного из аргументов и **приёмником результата** вычислений. Операции кодируются как правило в однооперандные инструкции (аккумулятор+операнд→аккумулятор). Такая архитектура характерная для многих CISC-процессоров (напр. Z80, x86 (i3,i7)).

Стековая архитектура

- **Стековая архитектура** — определяется организацией регистрового файла в виде стека, и косвенной адресацией регистров через указатель стека, который определяет положение вершины стека, операции производятся над значениями на вершине стека и результат кладётся также на вершину. Арифметические операции кодируются в нуль-операндные инструкции. Стековая архитектура является неотъемлемой частью MISC-процессоров

Введение в ядро КЭШ памяти

- Кэширование — это использование дополнительной быстродействующей памяти (кеш-памяти) для хранения копий блоков информации из основной (оперативной) памяти, вероятность обращения к которым в ближайшее время велика.
- Уровни КЭШ: L1-на каждом ядре, L2- общая для ядер раньше была на плате процессора, сейчас на кристалле МП, L3-дополнительная медленная

Расширение шины данных

- За 1 такт читается больше требуемой длины данных . Так, процессоры AMD K8 умели производить одновременно 64-битные запись и чтение, либо два 64-битных чтения за такт, AMD K8L может производить два 128-битных чтения или записи в любой комбинации. Процессоры Intel Core 2 могут производить 128-битные запись и чтение за такт.
- В итоге процессор может реже читать/писать память, выполняя несколько операций по результатам одного чтения из ПЗУ/ОЗУ.

Упреждающее чтение и фиктивное выполнение

- Процессор при свободной шине ОЗУ считывает в КЭШ из ОЗУ наперёд несколько (иногда десятков, сотен) команд. При выполнении команды извлекаются из КЭШ обычно за 1 такт.
- При появлении команд ветвления, фиктивно выполняются команды по разным веткам ветвления, пока реально не выполнится команда, результат которой определит реальный путь ветвления. Результаты выполнения (накопленные в параллельных регистрах процессора) ложных веток отбрасываются, а реальной ветки принимаются как выполненные за 1 такт.

Задачи и блоки ядра

В течение каждого цикла команды **ЦП** выполняет много управляющих функций:

- 1) помещает адрес команды в адресную шину памяти;
- 2) получает команду из шины ввода данных и дешифрует ее;
- 3) выбирает адреса и данные, содержащиеся в команде; адреса и данные могут находиться в памяти или в регистрах;
- 4) выполняет операцию, определенную в коде команды. Операцией может быть арифметическая или логическая функция, передача данных или функция управления;
- 5) следит за управляющими сигналами, такими как прерывание, и реагирует соответствующим образом;
- 6) генерирует сигналы состояния, управления и времени, которые необходимы для нормальной работы УВВ и памяти.

Обычно каждую функцию ядра выполняет блок ядра.

Выполнение операций (логических, целочисленных, с плав. запятой) осуществляют **исполнительные блоки**

Уменьшение тактов на одну команду

- В ранних МК (в МК Intel 8051) на 1 простейшую команду требовалось 3 цикла (чтение, декодирование и исполнение, запись результата) по 4 такта. Итого команда выполнялась за 12 тактов. При частоте 12 МГц тот процессор имел производительность 1 MIPS а его современные аналоги имеют при той же частоте 12 MIPS и более.
- Затем при введении RISC MISC простые команды (логические, целочисленные + - потом * и /) научились выполнять за 1 такт
- Сейчас в ядрах топовых RISC и MISC МП и МК любые команды, в том числе и с плав. запятой выполняются за 1 такт (и в среднем менее чем за 1).

Конвейер

- Процессор, выполняющий каждую команду за четыре такта (выбор команды, декодирование и выбор операндов, исполнение, запись результата) обеспечивает работой исполнительное устройство только в такте «исполнение». Для увеличения столь низкой (25 %) эффективности используется конвейерная организация: такты последовательно выбираемых команд совмещаются во времени так, что в каждый период одна из команд обязательно находится на этапе «исполнение», загружая исполнительское устройство.
- В итоге за 1 такт выполняется до 4 (и более) команд. В ARM CM3 в среднем 1,25... 1,5 команды за такт.

Реализация конвейера с 4-х тактными инструкциями

- | | T1 | T2 | T3 | T4 | T5 | T6 | |
|------|--------|---------|------------|------------|------------|------------|-------------------|
| • И1 | Чтение | декодир | исполнение | запись | | | |
| • И2 | | Чтение | декодир | исполнение | запись | | |
| • И3 | | | Чтение | декодир | исполнение | запись | |
| • И4 | | | | Чтение | декодир | исполнение | запись |
| • И5 | | | | | Чтение | декодир | исполнение запись |
| • | | | | | | | |
- В итоге за один такт (напрмер T4) выполняется одновременно 4 инструкции=команды.

Суперскалярные архитектуры

- **Суперскалярность** — архитектура вычислительного ядра, использующая несколько декодеров команд, которые могут **нагружать работой множество исполнительных блоков**. (в простом конвейере- по одному блоку)
- **Суперскалярность** -Способность выполнения нескольких машинных инструкций за один такт процессора. Появление этой технологии привело к существенному увеличению производительности

Многоядерность

- Вершина суперскалярности, **когда всех компонентов ядра** (не только исполнительных блоков а также узлов чтения, декодирования, записи, синхронизации, прерываний и т.д.) **более 1** и можно выделить автономные ядра с полным набором компонентов ядра, которые могут работать автономно и независимо друг от друга

VLIW-архитектура

- Архитектуры VLIW (Very Long Instruction Word — Очень Длинное Слово Команды).
- Отличаются от суперскалярной архитектуры тем, что решение о распараллеливании принимается не аппаратурой на этапе исполнения, а компилятором на этапе генерации кода. Команды очень длинные, и содержат явные инструкции по распараллеливанию нескольких субкоманд на несколько устройств исполнения. Элементы архитектуры содержались в серии PA-RISC. VLIW-процессором в его классическом виде является Itanium (и его аналог Эльбрус-3).
- Исследования показывают, что VLIW DSP (TMS320C6x фирмы Texas Instruments) выполняет мультимедийные задачи в 0,63 – 9 раз быстрее, чем суперскалярный процессор Pentium II без μ SIMD-расширения на той же тактовой частоте

CUDA

- **CUDA** (англ. *Compute Unified Device Architecture*) — технология GPGPU (англ. *General-Purpose computing on Graphics Processing Units*), позволяющая программистам реализовывать на языке программирования Си алгоритмы, выполнимые на графических процессорах ускорителей GeForce восьмого поколения и старше (GeForce 8 Series, GeForce 9 Series, GeForce 200 Series), Nvidia Quadro и Tesla компании Nvidia. Технология CUDA разработана компанией nVidia.
- В современных графических процессорах 128-256 ядер! Которые можно использовать для обычных вычислений, когда плата не сильно загружена графикой.